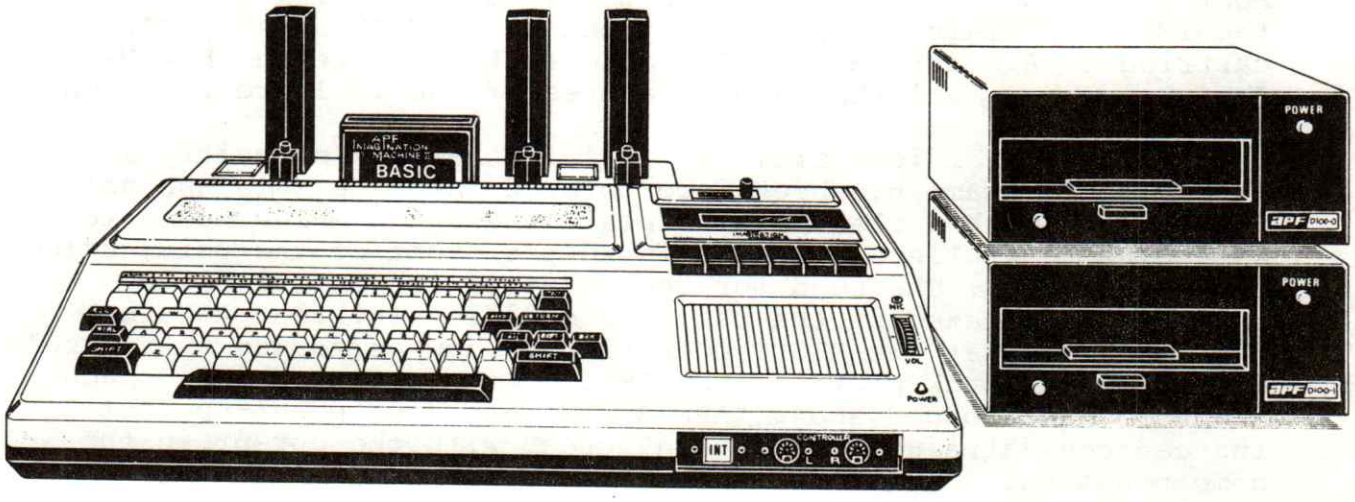
# ROLA DATOR

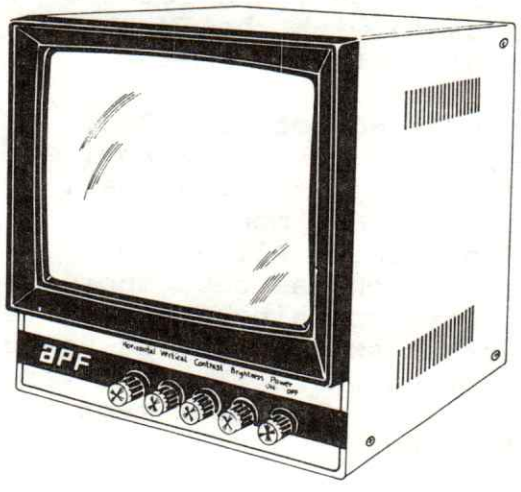
# A GENERAL INTRODUCTION




APF™ IMAGINATION MACHINE

## THE IMAGINATION MACHINE (IM-II) SYSTEM II

Exciting new small business/personal computer. User programmable in basic and uses APF software on cassette or mini-floppy diskette. The IM-II SYSTEM II comes complete with 14K ROM, 27K RAM (expandable to 75K RAM), Dual Floppy Disk Drives (D-100-A), Parallel Printer Interface (PI-100) and a Building Block (BB-2) Expansion Box with a Floppy Disk Interface (FI-100).




## TVM-10 — 9" VIDEO MONITOR

TVM-10 — 9" VIDEO MINITOR
- Built to Highest Standards • High performance
- 18 transistors • 16 diodes • H 9.6" x W 9.1" x D 10.1"
- For CCTV Camera, or Computers.

## THE APF IMAGINATION MACHINE II (IM-2)

The IM-2 is a logical development of the IM-1. The circuitry is all very similar to that of the IM-1. Modifications have been made in a number of important areas, however.

At present, the IM-2 uses the same LEVEL I BASIC as the IM-1. Most of the software written for the IM-1 will still run on the IM-2. The main exceptions are the BACKGAMMON and BASEBALL cartridges. All current APF Cassette software runs on the IM-2. Obviously, then, all the major features of the IM-1 are retained.

The appearance differs from the IM-1 in that the circuitry in the MP1000 TV game has been incorporated into the main cabinet and covered with a flat plastic plate where the MP1000 used to sit. The controllers now plug in to the front of the unit, using 8-pin DIN plugs. The need for the reset switch on the MP1000 has been eliminated by the use of a modified Internal ROM in the IM-2, so that the RESET operation is now performed by a single switch on the back plate of the IM-2. As a result of this change the screen no longer shows "APF BASIC. etc." on power-up, but instead goes straight to a green screen with the cursor in the home position.

Another improvement brought about by the new Internal ROM is in the area of machine language programming. If the IM-2 is powered-up without the BASIC Interpreter Cartridge inserted, the computer is in the Monitor/Debug mode. Instead of the normal block cursor, a "greater than" sign and a blinking cursor appear in the home position. In this mode, which can also be accessed by a CALL command from normal BASIC mode, the programmer is able to examine, modify and originate machine code programs. Two extra commands - V nnnn and nnnn - are available.

The RAM memory expansion of the IM-2 is increased to a maximum of 24K user RAM, when using LEVEL I BASIC. This increase is effected by inserting the R16-K cartridge in the BB-4. Basic configuration of the machine is still 8K user RAM, but now there is 3K screen memory available, instead of 1K in the IM-1, making a total of 11K. The R8-K cartridge of the IM-1 is not usable with the IM-2. All other peripherals from the IM-1 are usable with the IM-2.

The video ouput of the IM-2 is NTSC encoded composite video. There is an RCA jack on the back plate, and a cable is provided for connection to monitor. Also on the back plate is a small slide switch which enables the user to use a B/W monitor by cutting the chroma signal from the video output. All audio is now played through the built-in speaker, which has been increased in size to 3". The front plate of the unit contains a new switch marked "INT", which is a hardware interrupt switch, and is useful for stopping execution of any process during which the computer is not scanning the keyboard, e.g. during an input statement or while a program is being loaded onto tape.

The power supply requirements of the IM-2 are different from the IM-1, and a new model of power supply is provided. This has selectable inputs of 110,200, 220 and 240V AC 50/60Hz and outputs of 110VAC (for the monitor), 15VAC x 2 (for the disk drives) and 15VAC for the IM-2.

The back plate on the IM-2 also holds an audio output socket, enabling use of an earphone or connection to an external amplifier.

## LEVEL II BASIC

In July/August of 1981, APF will release a new language for use with the IM-2 - APF LEVEL II BASIC will come in cartridge form and will be a straight replacement with the existing LEVEL I cartridge. No other changes or modifications will be necessary to convert the unit to LEVEL II operation. No price has been decided on yet.

LEVEL II BASIC will,to a large degree, be compatible with MICROSOFT Level 2 Basic (as used in APPLE II and TRS-80). It should be possible to type a program written for one of those machines straight into the IM-2 and RUN it, with little or no modification.

When using LEVEL II, the IM-2 will be able to address up to 72K user RAM. This expansion will be made possible by future release of R32-K and R64-K cartridges. This is made possible by using a "bank select" method to address all RAM beyond 24K. LEVEL II will also make use of the additional screen memory of the IM-2, with new graphics capabilities.

The Disk Operating System (DOS) with LEVEL II will be on diskette, and will be loaded into memory. Four disk drives can be addressed.

Full details of the new language will be released as soon as possible.

## THE APF IMAGINATION MACHINE (IM-1)

The IM-1 is a powerful microcomputer based on a programmable TV game. The TV game (MP-1000) sits on top of the MPA-10 computer companion module and is joined to it by a special J-shaped connector. The MP-1000 contains the MPU (MC6800) as well as the Internal ROM (2K) and the Screen RAM (1K). The MPA-10 contains the main RAM (8K). Thus, whilst the MP-1000 can function alone as a sophisticated TV game, the MPA-10 serves no purpose on its own. In future, any reference to an IM-1 means an MP-1000 joined to an MPA-10 with the J-connector.

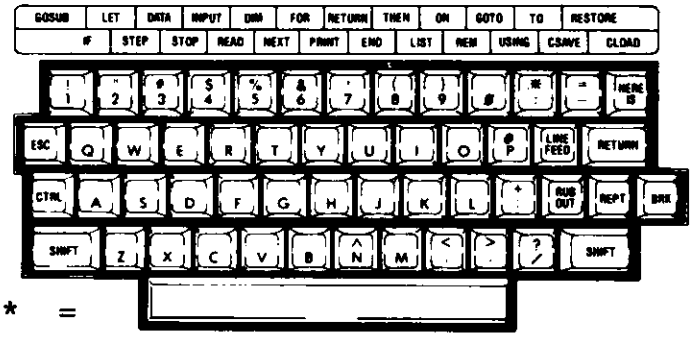
### IM-1 MAIN FEATURES

* RAM: 9,000 Bytes expandable to 17,000 Bytes
* ROM: 14,000 Bytes
* KEYBOARD:
  - *53-key stepped typewriter-style keyboard
  - *Two game-style hand controllers, each with joystick, fire button and 12-key numeric pad
* SCREEN FORMAT:
  - *32 characters x 16 lines (512 total) alphanumerics in 3 colour modes including inverse.
  - *Low resolution graphics - 64X32 in 8 colours fully-intermixable with alphanumerics
  - *High resolution graphics - Mode #1 128x192 in 4 colors
    - Mode #2 256x192 in 2 colors

    Both high resolution graphics modes are implemented as "object defined" systems.
* TAPE DECK:
  - *High quality built-in two-track cassette deck
  - *Uses standard Phillips-type cassettes
  - *Digital recording and playback at 1200 baud rate
  - *Speaker, volume control and microphone socket built-in
  - *3 digit tape counter
  - *Computer controlled motor
* BUILT-IN SOUND SYNTHESISER:
  - *Music synthesiser with 3 full octaves including flats and sharps
  - *Special effects capability
* LANGUAGE:
  - *ROM-resident BASIC language
  - *Full error commentary system
  - *Easy access to machine language
  - *Unique entry system of BASIC command words
* SOFTWARE:
  - *Available on cassettes, ROM cartridges and diskettes
  - *Full range of entertainment, educational and home management programs. Many programs are completely menu-driven

## KEYBOARD

53-key ASCII keyboard. Keyswitches are "break contact" type.



Letters
A-Z upper case only

Numerals
0-9

Punctuation
. , : ; ! ? " '

Symbols
# $ % & @

Mathematical Symbols
( ) [ ] / < > + - * =

Special Keys
RETURN
SPACE BAR
SHIFT - accesses upper symbol on keys as marked
BRK - (BReaK) stops program execution and returns control to user.
REPT - (REPeaT) simultaneous depression allows any key to print continuously on the screen, up to maximum of 128 characters.
RUB OUT - backspaces and erases last character on screen.
ESC - (ESCape) produces symbol on screen. Used for editing.
CTRL - (ConTRoL) when pressed, allows user to enter BASIC keywords (printed on plate above keyboard) by simultaneously depressing the corresponding key in the top two rows.
HERE IS - Not used in current version of BASIC
LINE FEED - Not used in current version of BASIC

All keys are software controllable.

## Game Controllers

Two game controllers (LEFT and RIGHT):

| | |
|---|---|
| *JOYSTICK | Four-position switch type |
| *FIRE BUTTON | Large push-switch |
| *KEYPAD | Numerals 0-9, CL (clear) and EN (enter) |

Both controllers are software controllable.

## BUILT-IN CASSETTE DECK

The built-in tape system has a number of unique features.

* The data transfer rate is 1500 baud. This means faster saving and loading.
* The system employs saturation recording on the digital track for maximum reliability.
* The record/playback head is a stereo unit. In this application one track is used for digital data, and the other is used for audio. The audio track is played back through the built-in speaker while the digital data is loading into the memory. All APF cassette software features an audio track, and the user can record his own audio through the built-in microphone socket.
* The tape system, upon the command CSAVE, automatically saves the entire contents of the user memory (RAM). Therefore all program statements and dimensioned variables are saved. This allows cassette-based filing systems.
* The first 512 bytes of information stored to the tape are always the contents of the screen. This information is then automatically fed back to the screen at the start of the loading operation. The data can be alphanumerics or graphics.
* No level, volume or tone adjustments are ever necessary. Tape loading and reading has never been this reliable!
* The motor is fully controlled by the computer. Automatic

## BUILT-IN SOUND SYNTHESISER

This built-in feature allows the computer to play music with a full three octaves, including sharps, flats, stays, pauses and so on. And programming couldn't be simpler:

```
Example -   10 MUSIC"/1/2/3/4/5/6/71234567*1*2*3*4*5*6*7"
```

This line plays the entire three octaves. Flats, sharps and so on are easy too:

```
Example -   20 MUSIC"+1+2+3+4+5+6+7"
            30 MUSIC"-1-2-3-4-5-6-7"
            40 MUSIC"100 200 300 400 500 600 700"
```

Line 20 plays all the sharp notes in the middle octave. Line 30 plays all the flat notes in the same octave. Line 40 plays the middle octave, but makes each note longer and puts a pause in between each note.

Two special sound effects are available too. These have been specially incorporated to make space games sound really exciting. They are accessed by simple CALL statements. CALL 17006 creates a small "BEEP" noise. CALL 17026 creates a combination whistle and "BEEP".

## VIDEO FORMAT and GRAPHICS

Alphanumerics:
The screen format for alphanumeric characters is 32 characters by 16 lines. Only upper case letters are available. 64 ASCII characters are available. These are all printable on the screen in the following colours:

1. Light green letters on dark green background (Normal mode)
2. Dark green letters on light green background (Reverse video mode #1)
3. Black letters on orange background (Reverse video mode #2)

Scrolling is automatic.

Low resolution graphics:
The screen is divided into 512 boxes (32x16). Each box can then be divided into four and displayed in any one of eight colors. Only one color is possible in each box, and any quadrant that is not colored will be displayed as black. Sixteen shapes are possible by different combinations of the four quadrants within one box. Theoretical resolution in this mode is, therefore 64x32.

A useful feature of this graphics system is that alphanumerics are fully intermixable with the graphics on the screen. This makes for very easy creation of "forms" on the screen for business or other applications.

The BASIC commands are very simple - COLOR and SHAPE are used to set the color and shape of the box, then PLOT will place that box in any desired location on the screen. Or use HLIN and VLIN to draw horizontal or vertical lines on the screen using the previously defined color and shape of box.

Example -
```
10 SHAPE=15
20 COLOR=3
30 HLIN0,31,10
```

This short program will draw a horizontal line across the screen from column 0 to column 31 using row 10. The line will be made up of solid red boxes, so it will appear as a straight, unbroken line.

The same effects can be created using POKE commands to place shapes in the desired locations on the screen. Similarly, PEEK commands make it very easy to write exciting action games in this graphics mode.

The colors available for low resolution graphics are - Light Green/ Yellow/Dark Blue/Red/White/Light Blue/Purple/Orange.

High Resolution Graphics:
For more interesting and exciting graphics, the IM-1 offers two modes of high resolution graphics:

Mode#1 - 128x192 in 2 color sets, each with 4 colors.
Mode#2 - 256x192 in 2 color sets, each with 2 colors.

Both graphic modes are implemented as "object defined" systems. An object or shape is defined, and then the screen map shows which object shape is placed in object boxes on the screen. This is analagous to the regular alphanumeric mode where the object shapes - the alphanumeric character set - is predefined in ROM. The main difference in the graphics mode is that the object shape is defined by the programmer in RAM.

In either mode #1 or mode #2 the screen map is divided into 32x12 boxes (384 total). Any one of the defined object shapes can be selected to be placed in any one of these 384 boxes. Each box is subdivided into 16 rows, each row with either 4 dots wide (mode#1) or 8 dots wide (mode#2). 512 bytes of memory are allocated for object shape definitions, each shape taking 16 bytes. Therefore a maximum of 32 objects can be defined.

In mode #1, each box can contain any combination of the four colors available in the selected color set. Two color sets are available.

In mode #2, each box can contain any combination of the 2 colors available in the selected color set. Two color sets are available.

To see these two modes of graphics in operation, look at the DEMO tape, which uses mode #1, and SPACE DESTROYERS, which uses mode #2. It is also possible to divide the screen into areas so that one part of the screen displays high resolution graphics, while another part is displaying normal alphanumeric characters. For example, see BASEBALL or BACKGAMMON.

PROGRAMMING

The IM-1 uses LEVEL I BASIC. This is APF's own version of the language, and is not directly compatible with any other version. The computer uses a full error commentary system.

GENERAL FEATURES

*Fixed four decimal place arithmetic
*No scientific or trigonometric functions
*All arrays must be dimensioned ( single and dual subscripts )
*Multiple statements are allowable, seperated by a colon
*Maximum line length is 128 characters
*Maximum line number is 9999, numbers must be integers
*Any number of nested loops are allowable
*All spaces, except those in REM statements or quotes, are ignored by the computer in memory. BASIC keywords are stored in memory as single characters.

*Number variables can be up to 13 digits long ranging from +999,999,999.9999 to -999,999,999.9999
*Letter variables can contain up to 100 characters. The maximum length of each letter variable must be defined with a dimension statement
*Variable names can be up to 5 characters long, but only the first two characters are used. Names cannot contain a keyword. The first character must be a letter. Letter variable names must end with a $.
*Maximum dimension for arrays is 999 (99 for strings):
e.g. DIM A(200), B(150,50), C(1,800) are legitimate
DIM D$(99) is the maximum allowable
but note DIM E$(320,10) is still OK

*Rules of precedence - operators have the following order of precedence, starting with the highest:

Negation
Exponentiation
Multiplication/Division
Addition/Subtraction
Relational Operators

When two operators of the same precedence occur, the expression is evaluated from left to right.

*Disk Operating System (DOS) is incorporated in Interpreter ROM.

COMMANDS

CLOAD
CSAVE
EDIT
LIST
RUN

INPUT/OUTPUT STATEMENTS

DATA
INPUT
PRINT
PRINT USING
READ
RESTORE
SPC
TAB

GRAPHICS/MUSIC STATEMENTS

COLOR
HLIN
MUSIC
PLOT
SHAPE
VLIN

DISK OPERATING SYSTEM (DOS)

CLOSE
DIR
INIT
OPEN
PRINT #
READ #
RUN
SAVE

STRING FUNCTIONS

ASC
CHR$
KEY$
LEN

ARITHMETIC FUNCTIONS

ABS
INT
RND
SGN

PROGRAM STATEMENTS

CALL
DIM
END
FOR...TO...NEXT
GOSUB
GOTO
IF...THEN
LET
ON...GOSUB
ON...GOTO
PEEK
POKE
REM
RETURN
STEP
STOP

OPERATORS

+
-
*
/
=
>
<

## Unique entry system of BASIC command words

The IM-1 features a unique entry system for BASIC keywords. Along the top of the keyboard is a plate printed with 24 words. Press the CTRL key and then any one of the keys in the top two rows on the keyboard and the corresponding keyword is automatically entered into the computer's memory and appears on the screen. The top row of words on the plate corresponds to the top row of keys, and the bottom row of words correspond to the second row of keys. Thus, to enter the word "PRINT" on the screen, just press the CTRL key and the "Y" key at the same time. PRINT appears on the screen. If you now press the RUB OUT key you will notice that the word disappears as if it were a single character. This is because the computer is storing that word in its memory as a single character. This feature allows the user to make far more efficient use of the memory size of the machine. Furthermore, even if you type the word out in full - P-R-I-N-T , the computer will still recognize it and put it in memory as a single character. The computer also ignores all spaces when the user is entering a program from the keyboard. You can type as many spaces as you like on the screen, but the computer will ignore them in memory (except for spaces in a REM or PRINT statement, and a PRINT USING image definition). This makes usage of memory space in the IM-1 even more efficient.

The following words are available from a single keystroke, using the CTRL key:

| | | | | | |
|---|---|---|---|---|---|
| GOSUB | DIM | ON | IF | NEXT | REM |
| LET | FOR | GOTO | STEP | PRINT | USING |
| DATA | RETURN | TO | STOP | END | CSAVE |
| INPUT | THEN | RESTORE | READ | LIST | CLOAD |

## MACHINE LANGUAGE PROGRAMMING

The IM-1 is also user-programmable in 6800 machine code. This mode is accessed by CALL 28672. Then, after entering a machine language routine, typing G8894 tells the computer to GO back to the entry point of BASIC. At this point you can write a BASIC program that calls the machine language routine. More than one routine can be written and called as necessary. All memory locations can be examined on the screen, including the BASIC Interpreter itself. Commands available are:

D nnnn - displays 16 bytes of memory beginning at hexadecimal address nnnn.
M nnnn - allows modification of the contents of a memory position.
G nnnn - the computer immediately begins executing the machine language routime at address nnnn.

(A disk-based Assembler/Editor and Disassembler system is available)

## SOFTWARE

Software is available in three forms for the APF IM-1:

1. Cassette
2. ROM cartridge
3. Diskette

### 1. CASSETTE SOFTWARE FROM APF

The programs available on cassette are divided into two main categories - a. Entertainment/Education and b. Home/Personal Management.

#### a. ENTERTAINMENT/EDUCATION

MA125 MUSIC COMPOSER/PLAYER PIANO - this program uses the full power of the built-in music synthesiser and graphics system to bring you a new dimension in sound. Make your own music, and print the words to the song on the screen at the same time. Or sit back and listen to one of the tunes already in the computer. And you can save you favourite tunes on cassette as well.

MA150 TYPING TUTOR - learn typing at your own speed with this unique program. The computer designs and changes the course to suit your ability - as you get better, it gets more difficult!

MA200 MATH TUTOR - this program is great fun for one or two players of any age! The computer sets mathematical problems for you, and you have to get the answers right in double quick time to beat your opponent (or the computer!)round the course to the finish line. An adult can play against a child, because the problems set for each player vary individually according to your performance.

MA225 PERCEPTION I - two players at any level compete in a game of recognition - shapes, numbers or letters. But watch out! As you get better, it gets harder!

MA325 ARTIST AND EASEL - this program uses the graphics of the IM-1 to maximum advantage. Using the joystick on the hand controller, paint a colorful picture on the screen. Store your favourite masterpieces on cassette so you can admire or improve them later. Or draw a series of pictures and tell the computer to string them together on the screen for your very own cartoon animation show!

MA375 THE WORD FACTORY - here's a fascinating game that really keeps you on the edge of your seat! You might recognize the picture on the screen, and you might think you know how to spell it. But can you catch the letters you need as they fly across the screen?

MA400 SPELLING DUEL - challenging and educational, this combination spelling and crossword game helps reinforce the players' English skills while it entertains.

MA550 JUMBLED UP THINGS - an exciting program that really tests the limits of your brainpower! Create words from jumbled up letters - against the clock! Test your memory, really test it, recalling the number that just flashed on the screen. Or try and recreate the picture on the screen in front of you - it isn't as easy as it looks!

MA575 SPACE DESTROYERS - Space Destroyers are no longer on the horizon. They're here! Our inter-galactic program pits your strategy and skill against an ominous force. Hundreds of SPACE DESTROYERS descend upon Earth. You must defend your station against these aggressors that stop at nothing. Armed with your laser-blaster, you carefully target your shots and fire. Again and again. Still they come on. Yet you must win this ultimate space fight, because if just one of them lands the Earth will be destroyed.

**********
*SPECIAL SPOTLIGHT ON.....*
**********

MA500 BASIC TUTOR - after months of hard work by writers and programmers at APF, the easy way to learn programming is here. The APF BASIC TUTOR program is complete. Complete and ready for all of you who want to write your own programs - but have been put off by those dreadfully boring text books. BASIC TUTOR is unique. It's a "hands-on" method. You learn to program at the Imagination Machine itself. (Not by having to plough through weighty volumes packed with theory but little practical application.)
This brand new method consists of 22 lessons. They are on two cassettes. Twelve lessons on one and ten on the other. You also receive a study guide, which does exactly that - guide you through the lessons as they appear on the screen of your monitor. But the important thing about the new BASIC TUTOR program is that you progress at your own pace. You set the tempo. There's a special monitoring system in Lessons 1 through 12, that will correct you if you make a mistake. You can stay with a lesson as long as you like. Until you are confident you've learned all the programming principles being taught in the lesson. There's even a special dictionary mode; you can refer to it at any time. It will give you the definition of many terms used in BASIC language.
The whole idea of BASIC tutor is to get you to understand how the computer thinks, to make you familiar with the "language" the computer speaks and understands.
In Lesson 1, you come to understand how a computer adds, subtracts, multiplies and divides.
By Lesson 3, you're dealing with loops, and understanding how the computer does it.
Lesson 7 gets you into using colour and the music starts in Lesson 11. Come Lesson 18, we'll show how to draw Snoopy on the screen. And when you've finished Lesson 22, you'll be an expert on the IM-1.
BASIC TUTOR is far and away the best way to learn programming. But it is not only for beginners. If you've tried programming but got "lost" along the way, then step straight into Chapter Two in the book and it will put you back on the programming tracks - give you new insight into just what you can do with a friendly computer and a well written program.
Even accomplished programmers will be helped by BASIC TUTOR. You can use the BASIC TUTOR Study Guide as a reference book. It's a library of programming ideas.
Each of the 22 lessons of BASIC TUTOR contains a program you can run on its own. Maths programs (which could help your children) games programs and practical financial programs.

.....YOUR LIFE WILL NEVER BE THE SAME AGAIN.....

b. HOME/PERSONAL MANAGEMENT

MA175 CHECKBOOK/BUDGET MANAGER - need a faster, easier way to balance your checkbook? Budget money? Keep tax records? You've just found what you need.
MA250 SPACE, SIZE AND SURFACE GUIDE - whether it's gallons of paint or pounds of grass seed, now you can accurately estimate amounts needed. And do money-saving cost comparisons of products, too.
MA275 PERSONAL BUSINESS MACHINE - calculate your financial status quickly and easily for up-to-the-minute accuracy. Compute loan amortization, interest and principal payments, interest rates, etc.
MA300 BUDGET MANAGER II - break out detailed expenditures (by day, by week, by purchase,etc.) in thirteen different categories. Food, medical, car,travel, plus nine more. Call up summary reports, too.
MA350 BAR CHARTS - one colorful barchart is good. But compared to what? Here's the way to have bar charts displayed side-by-side. Year-to-year and category-by category comparisons are easy with this program.
MA425 ELECTRONIC FILES - this sophisticated cassette-based filing system will keep track of all that information that is bulging out of the drawers in your study. Compile and maintain address lists, birthday lists, recipe lists, even a catalogue of the books in your personal library. Start keeping orderly, electronic track of things.
MA475 BILLBOARD/MESSAGE CENTRE - this program is one of our best. Using simple, easy-to-learn commmands you can put your name in lights on the screen, design a clever sign that flashes in time to the music, or create eye-boggling patterns on the screen. And all without needing to know any computer programming. This program is particularly effective for point-of-sale displays in retail stores.

## 2. ROM CARTRIDGE SOFTWARE FROM APF

Twelve plug-in ROM cartridges are available from APF. Simply unplug the BASIC Interpreter Cartridge and substitute one of these:

MG1001 CATENA - is APF's version of the popular computer game of Othello. Play against a friend or against the computer. Or even watch the computer play against itself so you can get the hang of it. But watch out if you're playing against the computer - it's been playing these game longer than you have!
MG1003 HANGMAN/TIC TAC TOE/DOODLE - three games on one cartridge! Hangman will keep the kids entertained for hours, and Tic Tac Toe (Noughts and Crosses) is great for any age. Doodle lets you draw pictures on the screen using the hand controllers to guide your "brush".
MG1004 BOWLING/MICRO MATCH - select the level of skill you want, then try and knock down those pins - it isn't as easy as it looks! Micro Match is a real brain-teaser, and you have to concentrate very hard to remember where that pair was on the table - the computer spreads out the "cards" on the screen, and you and your opponent have to turn them up two at a time to find the pairs. Frustrating fun!
MG1005 BRICKDOWN/SHOOTING GALLERY - Brickdown is an old favourite - keep the ball bouncing against the wall and try and break a hole through it as each bounce knocks out another brick. Shooting Gallery will test your reflexes to the limit. You have to be very quick on the FIRE button to hit the moving target.
MG1006 BASEBALL - the game will never be the same! Now you can get all the action of the big game right in your home. Move your back-fielders, pitch a real banana ball, or try for a strike with a fast pitch. Then watch your players actually run across the screen to catch the ball! It sure beats Sunday morning in the gym!
MG1007 BLACKJACK - call it Blackjack, Vingt et Un, or Twenty One. It is still the best card game ever played, and this is the best full-graphics version you will ever see, or lose money to!
MG1008 BACKGAMMON - now put your thinking cap on. Moving your pieces round the board on the screen, you've got to beat your opponent or the computer. The honor of the club is at stake!
MG1009 CASINO I -(ROULETTE/KENO/SLOTS) - You can hear the wheel spinning, the ball clattering round, and you have all your money on the zero - will your gamble pay off? Try your hand at Keno (a close relative of Bingo), or play the poorman's slot machines.
MG1010 UFO/SEA MONSTERS/BREAK IT DOWN/REBUILD/SHOOT - five great games on one cartridge, all requiring a good eye and a steady hand!
MG1011 PINBALL/DUNGEON HUNT/BLOCKOUT - get all the fun of the arcade with Pinball. Try and outguess your opponent in Dungeon Hunt. Then pit your wits against him in Blockout, a mis-leadingly simple game of strategy for two.
MG1012 BOXING - ringside action! Spar with Slugger Sam, or go for a knockout against Horrible Harry. But watch out - these guys are veterans of the ring!
MG1013 SPACE DESTROYERS - NOW AVAILABLE AS A CARTRIDGE!

## 3. SOFTWARE PACKAGES ON DISKETTE FROM APF

### BP-30 ACCOUNTS RECEIVABLE

Requires: IM-1, BB-4, FI-100, D-100-0/1, SI-232/PI-100,R8-K,printer

APF's Accounts Receivable system uses standard accounting procedures for maintaining a file of customer information and transactions, preparing monthly statements, up-dating accounts, and producing an aged report using 30-60-90 day standards.
Ant business that maintains customer records will find this system easy to use and a valuable tool for minimizing paper-work. Customer information and transaction is easily recorded on diskettes, and the optional daily report printout is very useful. All additional functions can be performed by the computer on request.

### SY-20 ASSEMBLER/EDITOR

Requires: IM-1, BB-4, FI-100, D-100-0

Now there is a full Assembler/Editor available, to let experienced programmers really explore the powerful features of the IM-1. One important feature is the full screen editor. This enables the user to insert or delete a character or a line, page forward or backward. It will also print out or display symbols, syntax errors and address locations as it assembles your machine language programs.

### SY-30 BACKUP

APF's Backup program enables the user to make complete copies from any single-sided diskette onto a formatted (clean) diskette using either a single drive or dual drive. A strong feature of the Backup program is that it verifies each file that it copies, which ensures program accuracy.

### SY-40 DISASSEMBLER

This program will give a disassembly of memory contents in MC6800 mnemonics. A very useful package for serious programmers.

### BP-40 MAILING LIST

USA version with Zip codes and 3-letter abbreviations for State.

### BP-50 GENERAL LEDGER

Designed to match the BP-30 Accounts Receivable system. Available July 1981.

# ERROR MESSAGES

**The following is a list of error messages. If they occur during a program statement, the statement number as well as the message is displayed.**

**ARITHMETIC OVERFLOW**
The result of a computation is greater than 999999999.9999. This problem can be eliminated by using the IF statement to test and limit the sizes of the variables in the equation before doing the computation.

**DIMENSION**
Something is wrong in Dimension Statement (size is zero or greater than 99, etc.).

**DIVISION BY ZERO**
The result of dividing by zero is undefined. The computer cannot proceed. This problem can be prevented by checking the value of a divisor with an IF statement before doing the division.

**EXPRESSION**
The expression is not properly formed. Examples are PRINT PEEK 123 – parenthesis was not used around expression of a function. PRINT 234 + 3) – missing an open parenthesis. PRINT 1 ← 2 – improper operator symbol in an expression.

**EXPRESSION MISSING**
An expression is expected to be located in a statement but is not found.

Example: A = (Return Key)
For H = (Return Key)

**EXPRESSION TOO LONG**
You did nothing wrong. All computer languages have limits, and you just ran up against one of ours. This will occur if an expression has more than 8 nested brackets or parenthesis.

**FOR – NEXT**
A NEXT statement was executed when no FOR statement had been executed. Check your program. You have forgotten to supply the FOR statement or the logic of your program is incorrect. Somehow you branched to a NEXT statement without first going through its FOR statement. Fix the logic of your program before continuing.

Example: **10 NEXT J**
**RUN**

**IF – THEN**
The comparison in the IF statement is wrong. For example, IF A*B THEN 100. Fix the statement before running the program.

**ILLEGAL MASK SIZE**
The format defined for a PRINT USING statement is incorrect. For example, there may be more than 8 "#"s to the left of the decimal place, or more than 4 to the right. Correct the PRINT USING format, called a mask, before running the program.

**ILLEGAL VARIABLE**
This is a catchall error message when the interpretor finds something wrong with a variable. Some examples are:

A variable name starts with an illegal symbol – i.e., a number (1A = 3). You have used more than 26 variable names. You used a dimensioned variable name that was not dimensioned.

**MEMORY FULL**
You ran out of memory. This will usually occur in a DIM Statement. If so, then reduce the allocation in a DIM statement or shorten your program.

Memory storage is as follows:
1. A numeric variable takes 7 Bytes. A dimensioned numeric array takes 7 Bytes for each element.
   Example: DIM H (365) takes 7*365 Bytes.
2. Each line takes 2 Bytes for a line # plus 1 Byte for an end of line symbol.
3. All keywords (i.e., – PRINT, FOR, etc.) take 1 Byte.
4. Actual allowable user storage is 7166 Bytes.

A second reason for Memory Full can be due to your program's continuous execution of a FOR statement with no NEXT statement, or continuous executing a GOSUB with a RETURN.

**NO – GOSUB**
A RETURN statement was executed without a GOSUB having been executed. You have a logic error in your program. Make sure you do not use a GOTO or fall into a Subroutine. Fix the problem before running the program.

**NO LINE #– REFERENCED**
You are trying to GOTO a line number which doesn't exist in your program. Correct the statement before running your program again.

**PRINT DELIMETER**
There is an error in your PRINT statement. Check to see that each of the items being printed are separated by a comma or semi-colon. Correct the statement before running your program.

**QUOTE MISSING**
The right quote in a String Constant is missing. Correct the statement before running the program again.

**READ – DATA**
Either you have executed a READ statement when there is no DATA statement, or there is an error in the DATA statement. If there is an error in the DATA statement, correct it before running the program again.
If you are executing a READ statement, then either you forgot to include a DATA statement or you have tried to READ more variables than you have data in DATA statements. Make sure you have supplied all of the data in the DATA statements, or that you haven' executed a READ statement more times than you had intended.

**WHAT**
This is a catch-all. It usually means that one of the items in the line should have been a keyword but wasn't recognized as such. Carefully check the line and correct all errors before running the program again.

**#> 999999999.9999**
You have a constant which has more than four decimal places or which is greater than 999999999.9999 or less than – 999999999.9999. You will have to change your program so, that numbers outside of this range are unnecessary.

**The following are possible error messages that can occur when using disk statements.**

**FILE ABSENT**
File name is not in the disk directory. Be careful that the names always match exactly. If a name was opened as a value of a string variable then always use a string variable when using that name.

**16 FILE LIMIT**
If there are sixteen file names already in the directory and you want to open a new name, you must either delete a named file or use a new diskette.

**DISK FULL**
If all space on diskette is used there may not be sixteen file names because a single data file can use up to half the disk space.

**WRITE ERROR**
1. Usually occurs during a format when you are trying to write directory, if no address marks are found check all connections and power supplies.
2. After writing a read check is done. There is an error if what is read does not corralate with what you wrote.

**READ ERROR**
1. A CRC (Cyclic Redundancy Check) error was detected or a lost data bit.
2. Writing or reading to an unopened data file. Data files must be opened before they can be read from or written to.
3. Trying to open one of the four data files that is already opened. It is a good practice to have the first statement of a program close all files (0.3).

**RECALIBRATE NOT SET**
1. Diskette not engaged correctly.
2. Diskette not formatted.

**EXPRESSION**
1. Usually syntax of the statement is incorrect. Most common is in print or read to data file, or the semi-colon is not placed correctly.
2. Writing a "PRINT" or "READ" statement in the format for a sequential file that was opened as a direct file or vice versa.

**RETRY**
Tried five times to read from diskette. Reinitialize system and re-load diskette.

Use as a handy reference guide or as an index to where detailed explanations are contained. The 'reference' column refers to the APF BASIC TUTOR (MA-500)

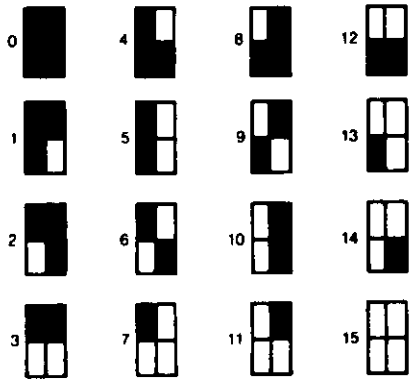
*Note:* (EXPR) means a numerical value or the result of a numerical expression
(STR$) means string value.

| TERM | EXAMPLE | MEANING | REFERENCE |
|---|---|---|---|
| **A** | | | |
| ABS(EXPR) | PRINT ABS (X)<br>P = ABS (Y) | A function that returns with the absolute value of an expression. A number's absolute value is its value without a (+) or (—) sign | — |
| ARRAYS | DIM X(10)<br>DIM Y(6,20)<br>DIM S$(6)<br>DIM P$(9,50) | A collection of values grouped together by a common name. Must precede an array with Dimension Statement. Each member or subscript of an array can have a different value. | Lesson 14 |
| ASC (STR$) | PRINT ASC ("A")<br>D = ASC (C$)<br>PRINT ASC (A$(4)) | Converts a character string variable to the ASCII integer code. Will return with first character of STR$ unless specified. | Lesson 8 |
| ASSIGN-MENT | PRICE = 1.3 | Statement that gives (assigns) a value to a variable. Value can be a number, result of a calculation, a name or a word. | Chapter 4 |
| **B** | | | |
| BREAK | – | System command which allows keyboard interruption of program. Tells computer to return to immediate mode. | Chapter 3 |
| **C** | | | |
| CALL EXPR | CALL 17046 | Tells computer to branch to MC6800 machine language routine which starts at memory location (EXPR) | Lesson 8 |
| CHR$(EXPR) | PRINT CHR$ (83)<br>A$ = CHR$ (48) | Used to obtain the single character represented by ASCII number code in ( ). | Lesson 8 |
| CLOAD | – | Turns on tape deck motor and audio. Will load program from tape to memory. | Chapter 5 |
| COLOR | Color = 4<br>Color = A | Sets color to be drawn on screen in either PLOT, HLIN or VLIN statements. Eight colors are available. | Lesson 7 |
| CSAVE | – | Save to tape. First saves 512 bytes which will be loaded to screen, then saves all of RAM memory. | Chapter 7 |
| **D** | | | |
| DATA | 300 DATA 1,2,3,4,5<br>310 DATA "JAN,"<br>"FEB," "MAR" | Specifies data to be read with a READ statement. The list may be constants or strings. List elements must be separated by commas. Data items will be read sequentially. | — |
| DELIMETER | PRINT "HELLO",<br>"OUT THERE"<br>PRINT "HELLO";<br>"OUT THERE" | Sets the boundaries between items we wish to be printed. BASIC delimeters are, and; | Chapter 3 |
| DIM | DIM A$ (10) | Tells computer to reserve space in its memory for letters, words or tables of numbers. Sets all initial values of string variables to null. All string variables must be dimensioned. | Lesson 8 |
| **E** | | | |
| EDIT | 100 INPUT A$: PRINT IN$<br>EDIT 100 [A$ [IN$<br>LIST<br>100 INPUT IN$: PRINT IN$ | Used to change portion of line in program. Used with ESC key. | Chapter 7 |
| END | 50 IF A = 100 GOTO 70<br>60 PRINT B$<br>70 END | Terminates program execution and returns cursor to screen. | Chapter 3 |
| EQUATION | COST = PRICE*NUMBER | The procedure or steps followed to achieve a result. | Chapter 4 |
| ERROR MESSAGE | WHAT<br>PRINT DELIMETER | Computer tells you that you've done something wrong. If they occur during program statement, statement number and error message is displayed. | Chapter 3 (plus appendix A) |
| **F** | | | |
| FLOW CHARTS | – | Visual programming aid that takes program statements and shows them in sequence via pictorial representation. | Chapter 4 |
| FOR | FOR L = 1 to 10<br>FOR X = Y to Z | Opens a FOR – NEXT Loop and initializes value of variable and increments it by amount in expression each time NEXT statement is executed. (see STEP) | Lesson 3 |
| FORMULAS | PRINT D/T; "MILES PER HOUR" | Formulas are solved directly and you may use any formula in a program. | Lesson 17 |
| **G** | | | |
| GOTO | 50 IF A = 20<br>GOTO 100 | Allows change in normal sequence of statement execution. Program jumps back or goes to specified line number. | Chapter 3 |
| GOSUB | 60 GOSUB 100 | Allows transfer of program to subroutine that can be written once but used many times in program. (see RETURN) | Lesson 16 |
| **H** | | | |
| HLIN | SHAPE = 15:COLOR = 4<br>HLIN 6, 10, 4<br>HLIN X1, X2, Y | Draws a horizontal line/or shape at a specified row on screen from column X1 to column X2 in Row Y (see VLIN) | Lesson 18 |
| **I** | | | |
| IF | IF A + B = C THEN<br>PRINT "YOU WIN" | Tells the computer to test something. It tests a condition that follows the word IF. If it is true, then BASIC will execute statement following word THEN. If false, program will continue to next step number. | Lesson 2 |
| IMMEDIATE MODE (no statement #) | PRINT "HI" | Tells computer to execute command at once rather than store it for later use. | Chapter 3 |
| INPUT | INPUT "PRICE", P<br>INPUT "NAME", A$<br>INPUT "COST", C,<br>"MILES", M | Allows assignment to a variable by entering value from keyboard. | Lesson 1 |
| INT(EXPR) | INT (3.14) = 3<br>PRINT INT (7.1*8.3)<br>56 | Returns the integer portion of the expression. The expression can be a number, a numeric variable or a calculation. | Lesson 6 |
| **K** | | | |
| KEY $(0) | 10A$ = KEY$(0)<br>20 IF A$ = NULL$ THEN 10<br>30 IF KEY$(0) <> NULL$ THEN 30<br>40 PRINT A$ | Reads the computer keyboard and looks for an input. Key $ (1) reads right game controller Key $ (2) reads left controller. | Lesson 5 |
| **L** | | | |
| LEN (STR$) | DIM A$ (10)<br>A$ = "ABC"<br>PRINT LEN (A$)<br>3 Length is 3 | Returns # of characters (length) of the string variable | Lesson 8 |
| LET | 10 LET PRICE = 1.3 | Optional word in APF BASIC that can appear before variable name in assignment statement. | Chapter 4 |
| LIST | LIST<br>LIST 100<br>LIST 200,5<br>LIST 300, | System command to display program. Program will appear in numerical order starting with lowest step number. Can also call out specific program steps or list parts of program. | Chapter 3 |
| **M** | | | |
| MUSIC | MUSIC "1234567" | Enables computer to play music using numbers 1 through 7. | Lesson 11 |
| MULTI-STATEMENTS | 40 PRINT B: INPUT C<br>50 PRINT A: GOTO 100 | Allows more than one BASIC statement per line number. Must be separated by : | Lesson 2 |
| **N** | | | |
| NEXT | 200 NEXT I<br>300 NEXT A,B | Used to return program execution to FOR statement in FOR – NEXT Loop so that current value of variable can be increased until it exceeds end value. | Lesson 3 |
| NULL | 10 DIM B$(15),<br>NULL$ (15)<br>20 B$ = "ABC"<br>30 PRINT B$<br>40 B$ = NULL $<br>50 PRINT B$ | A character that is nothing and cannot be printed on screen. When strings are initially dimensioned, each position is assigned code for a NULL character. | Lesson 8 |
| **O** | | | |
| ON-GOSUB | ON A GOSUB 200, 400, 600 | Goes to a subroutine depending on a variables value | Lesson 15 |
| ON-GOTO | ON A GOTO 100, 240, 560 | Changes program flow depending on variable's value. There can be up to 9 statement numbers that GOTO can direct program to. | Lesson 15 |
| **P** | | | |
| PEEK(EXPR) | PRINT PEEK (512)<br>128 | Examines location in computer memory and returns the numeric value contained there. | Lesson 11 |
| PLOT | PLOT 2, 1<br>PLOT X, Y | Draws a shape on the screen in one of 512 boxes. Each box has a row and column number. | Lesson 7 |
| POKE | POKE 784,65 | Places a number in a memory location. Often used to change cursor pointer contents used in a print statement. | Lesson 12 |
| PRECEDENCE | PRINT 4 + 6/3<br>6<br>PRINT (4 + 6)/3<br>3.3333 | Set rules of which operations are done first in a mathematical expression. Rules of precedence can be offset by ( ) or [ ]. | Lesson 1 |
| PRINT | PRINT 2 + 3<br>PRINT "HELLO"<br>PRINT A$ | Tells computer to display on screen the results of arithmetic problem, something enclosed in quotation marks or a variable value. | Chapter 3 |
| PRINT USING | 10: ### ##<br>20 PRINT USING 10, 123 | Allows you to print values (number or strings) in a specific format that you can specify. | Lesson 10 |
| PRINT # | PRINT # 1 | This mode sends results of PRINT and LIST statements to a printer, if attached. | — |
| | PRINT # 2 | This mode transfers command to an external device or terminal, connected via S1-232. The keyboard is inactive. | |
| | PRINT # 0 | Restores normal mode after above modes. | |
| **R** | | | |
| READ | 100 DATA 37; BILL,<br>8, SAM<br>200 Read A, B$ | Reads data into specified variable from DATA statement. Values are assigned in sequence starting with the first value in first DATA statement. | — |
| REM | 30 REM THIS IS A TEST | Remark statement for adding title, comments or remarks to your programs. Computer ignores words in statement after REM. | Lesson 13 |

| TERM | EXAMPLE | MEANING | REFERENCE |
|---|---|---|---|
| RESTORE | 100 DATA 37, Bill, 8, Sam<br>200 READ A, B$<br>300 RESTORE<br>400 READ C, D$ | Resets the data pointer so that the next READ statement begins with the first DATA statement in the program. C will have value of 37. | – |
| RETURN | 150 GOSUB 170<br>160 PRINT C<br>165 STOP<br>170 C = RND (0) * 5<br>180 RETURN | When used with GOSUB – RETURN transfers program back to statement following GOSUB instruction | Lesson 16 |
| RND (0) | A = RND (0) *7<br>A = RND (0) *10 + 1 | Creates a random number between .00 and .99 | Lesson 6 |
| RUN | RUN | System command that tells computer to find statements it has in stored mode and execute them starting with lowest line number. Also initializes all variables back to zero. | Chapter 3 |
| **S** | | | |
| SHAPE | 10 SHAPE = 15 | Sets the shape to be drawn on screen in either a PLOT, HLIN or VLIN statement. There are 16 shapes available. | Lesson 7 |
| SPC(EXPR) | PRINT SPC (10); "YOU WIN" | Space function that is used to insert a specified number of spaces (blank positions) | – |
| STEP | For X=1 to 20 STEP 2<br>For X=20 to 1 STEP-1 | Used to specify the increment to be added to or subtracted from the current variable value in a FOR – NEXT Loop. | Lesson 3 |
| STOP | 50 IF A<10 GOTO 70<br>60 IF A>10 GOTO 80<br>70 STOP<br>80 PRINT A/B | Causes program to stop running immediately and return control to immediate mode. | Lesson 1 |
| STORED MODE | 40 INPUT "B", B<br>50 PRINT A*B | Statements which the computer stores for usage at a later time instead of taking immediate action and executing them at once. | Chapter 3 |
| STRING CONCATENATION | | Joining of two strings into one. | Lesson 8 |
| SUBSCRIPT | DIM X(10)<br>DIM Y(A) | Distinguishes each member of ARRAY. Must always be numbers or numeric value. Can be referenced by variable with numeric value. | Lesson 14 |
| SYMBOLS | COST = A*B | Short cut ways to designate variables | Chapter 4 |
| SGN (EXPR) | SGN (–6) = –1 | Returns with arithmetic sign of expression. Returns –1 if EXPR is negative, 0 if EXPR is 0, +1 if EXPR is positive. | |
| **T** | | | |
| TAB | 10 PRINT TAB (5); "HERE"<br>20 PRINT "WHY"; TAB(2); "DID THIS HAPPEN" | Moves printing to the right on screen, like a typewriter tab function. TAB ignored if cursor has already passed number of TAB spaces requested. | Lesson 16 |
| THEN | IF K = 1 THEN GOTO 20 | Second part of IF – THEN statement that will only be executed if conditions in IF portion are true. Otherwise, statement following THEN is ignored. | Lesson 2 |
| TO | FOR J = 1 TO 50 | Establishes the END value in the FOR – NEXT Loop | Lesson 3 |
| **V** | | | |
| VARIABLES | NAME$ = "BILL" (string variable)<br>PRICE = 123.45 (number variable) | Symbols used to represent values. Two types are numeric variables (such as PRICE) and string variables (such as NAME$) | Chapter 4 & Lesson 8 |
| VLIN X1, X2, Y | SHAPE = 15; COLOR = 4<br>VLIN 0, 15, 20<br>VLIN Y1, Y2, X | Draws a vertical line or shape at a specified row on screen from row Y1 to row Y2 in column X. | Lesson 18 |

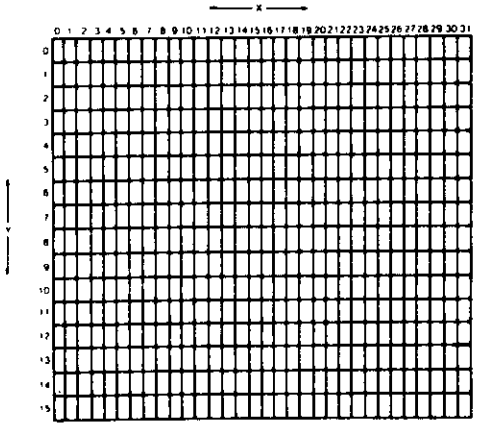
# GRAPHICS ( LOW RESOLUTION )

The shape table shows the number for each cursor shape.




Shape is set equal to a constant or expression. Its value is used in the next plot or line command. Modulo arithmetic is performed so shape can have a value greater than 15. Any cursor position can be any one of eight colors at any time by assigning it a color number from 0 to 7. Setting COLOR = expression determines the color to be used in the next line or plot command. Modulo arithmetic is performed so the expression can be greater than > 7.

0 = Light Green  2 = Dark Blue  4 = White  6 = Purple
1 = Yellow  3 = Red  5 = Light Blue  7 = Orange




# RESERVED WORDS

The following words cannot be used as or contained in variable names:

| | | | |
|---|---|---|---|
| ABS | FOR | NEXT | RUN |
| ASC | GOSUB | ON | SAVE |
| CALL | GOTO | OPEN | SGN |
| CHR$ | HLIN | PEEK | SHAPE |
| CLOAD | IF | PLOT | SPC |
| CLOSE | INIT | POKE | STEP |
| COLOR | INPUT | PRINT | STOP |
| CSAVE | INT | READ | TAB |
| DATA | KEY$ | REM | THEN |
| DIM | LEN | RESTORE | TO |
| DIR | LET | RETURN | USING |
| EDIT | LIST | RND | VLIN |
| END | MUSIC | | |

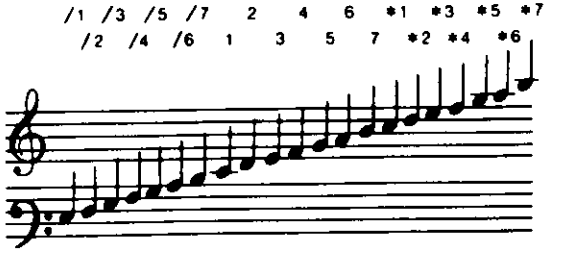
# SCREEN CODES

| Decimal | Will Appear On Screen | Decimal | Will Appear On Screen | Decimal | Will Appear On Screen | Decimal | Will Appear On Screen |
|---|---|---|---|---|---|---|---|
| 0 | @ | 15 | O | 32 | | 47 | / |
| 1 | A | 16 | P | 33 | ! | 48 | 0 |
| 2 | B | 17 | Q | 34 | " | 49 | 1 |
| 3 | C | 18 | R | 35 | # | 50 | 2 |
| 4 | D | 19 | S | 36 | $ | 51 | 3 |
| 5 | E | 20 | T | 37 | % | 52 | 4 |
| 6 | F | 21 | U | 38 | & | 53 | 5 |
| 7 | G | 22 | V | 39 | ' | 54 | 6 |
| 8 | H | 23 | W | 40 | ( | 55 | 7 |
| 9 | I | 24 | X | 41 | ) | 56 | 8 |
| 10 | J | 25 | Y | 42 | * | 57 | 9 |
| 11 | K | 26 | Z | 43 | + | 58 | : |
| 12 | L | 27 | [ | 44 | , | 59 | ; |
| 13 | M | 28 | \ | 45 | - | 60 | < |
| 14 | N | 29 | ] | 46 | . | 61 | = |
| | | 30 | ↑ | | | 62 | > |
| | | 31 | ← | | | 63 | ? |

| | | | |
|---|---|---|---|
| 64-127 | Same as 0-63 but reverse video. | 192-207 | Shape 0-15 in white color. |
| 128-143 | Shape 0-15 in green color. | 208-223 | Shape 0-15 in light blue color. |
| 144-159 | Shape 0-15 in yellow color. | 224-239 | Shape 0-15 in purple color. |
| 160-175 | Shape 0-15 in dark blue color. | 240-255 | Shape 0-15 in orange color. |
| 176-191 | Shape 0-15 in red color. | | |

# CALLS AND POKES

CALL 17046 – clears screen to black
CALL 34040 – turns tape deck motor and audio on
CALL 34061 – turns tape deck motor off
CALL 34138 – saves memory to tape
CALL 34225 – reads data to memory from tape
CALL 28672 – causes entry into machine language mode 1
CALL 18106 – enters machine language mode 2
(G8894 – returns to BASIC)
CALL 17006 – creates a small "beep" noise
CALL 17026 – creates a combination "whistle" and "beep"
POKE 40960,2: POKE 40961,0 – moves cursor to top left corner of screen.
POKE 24578,62 – turns motor on for audio recording through microphone
POKE 24578,38 – turns motor off after audio recording
POKE 24578,54 – stops audio track on tape from playing through speaker
POKE 8193,60: POKE 8194,158 – enters 128 x 192 graphics mode
POKE 8193,60: POKE 8194,222 – enters 256 x 192 graphics mode
POKE 8193,52: POKE 8194,30 – returns to 64 x 32 graphics mode
POKE 8193,60 – changes reverse video mode to black/orange
POKE 8193,52 – returns reverse video mode to green/green

# MUSIC

You can also play sharps and flats by typing a "+" or "–" sign in front of the note.

The number zero (0) is used to hold the previous note.

Leaving a space between numbers creates a pause in the music.

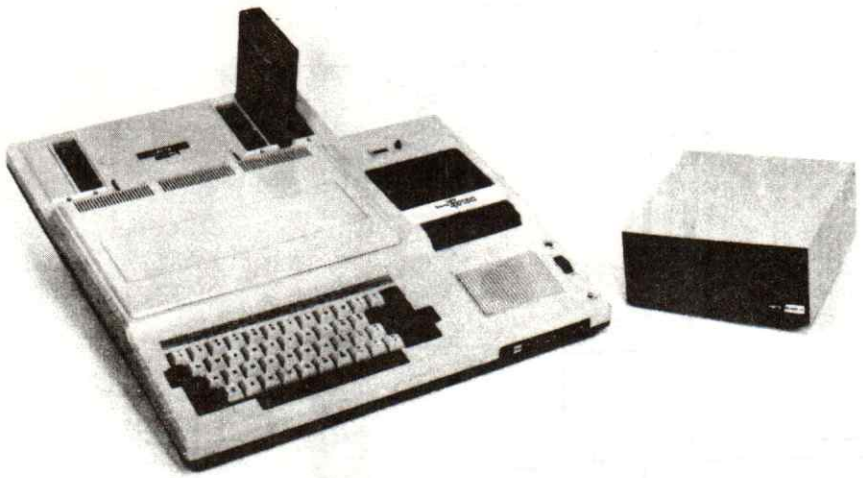
The sequence of numbers and notes to be played is placed inside a pair of quotes following the word **MUSIC.**

Example: **MUSIC "3212333" MUSIC "3212333000"**

**David H. Ahl**

CREATIVE COMPUTING

# APF Imagination Machine II



In 1978, APF Electronics, caught up in the video game craze, introduced the MP-1000. MP-1000 accepted plug-in cartridges of various games. Almost with one voice, most of the video game manufacturers a year later announced add-on keyboard units to make their video games into "real computers." Some of these were produced on schedule (almost), some were produced late (very), and most just quietly faded away. APF was in the first group having produced a keyboard unit almost on schedule. We salute them.

The keyboard/game unit combination is now available in a combined unit as the Imagination Machine I.

At the 1981 Winter CES, APF introduced an integrated version of the Imagination Machine (IM) aimed more at the small business market called the Imagination Machine II. First shipments were made in May 1981. Marty Lipper confided to me that consumer and dealer reception has been quite enthusiastic in Europe, hence APF is filling that pipeline first.

**Using the Imagination Machine II**

I had an opportunity to use an Imagination Machine II at the Summer CES in June. While this is not an in-depth evaluation, it will provide some idea of the capabilities of the IM II.

The IM II, unlike other popular computers, is built around the 6800 mpu.

The basic IM II unit houses the processor, 53-key full-stroke keyboard, dual-track cassette recorder (one track digital information, one track audio) and interface electronics for disk drives and a printer.

The IM II drives a monitor or color TV set. The display is 32 characters wide by 16 lines. There are 70 screen codes: 26 upper case letters, 10 numerals, 28 symbols and 16 graphics characters. The graphics characters divide each screen location into four elements thus, according to current convention, the screen resolution is (32 x 2) x (16 x 2) = 2048 pixels, a new low on the graphics totem pole. On the other hand, machine language access to print graphics is possible which gives the user the ability to manipulate 256 x 192 points (49,152 pixels) albeit with much less ease than low resolution shapes.

Although resolution is low, the graphics are easy to use. APF Level I Basic has PLOT, HLIN and VLIN commands. To put Shape 9 (a 2 x 2 checkerboard) at the center of the screen (X=15, Y=7) requires only the following simple command sequence:

```
SHAPE=9
PLOT 15,7
```

To color the shape any of seven colors, it is necessary to add a single line. For example, to color the shape red, type:

```
COLOR=3
```

However, to make it even easier to use colored shapes, screen codes from 128 to 255 represent each of the 16 shapes in each of the eight colors. Hence, a dark blue checkerboard would appear at the bottom right of the screen given the following commands:

```
SHAPE=169
PLOT 31,15
```

To plot a line of checkerboards across the center row (15) of the screen, one need only type:

```
SHAPE=9
HLIN 0, 31, 7
```

which draws a horizontal line (HLN) from X=0 to X=31 at Row Y(7). VLIN does the same for vertical lines.

The IM II has a one-voice music box built in with a three octave range. No complicated POKE commands are needed. Music is simply produced with the command MUSIC "nnn" in which nnn represents a series of notes represented by numbers. Duration is indicated by the number of 0's following a note value. Sharps and flats may be indicated with + and - symbols. Here are the first two bars of Haydn's Symphony #94 (second movement) with which I may have driven everyone batty at APF's press reception at Summer CES:

```
MUSIC"10 10 30 30 50 50 3000
40 40 20 20 /70 /70 /500000"
```

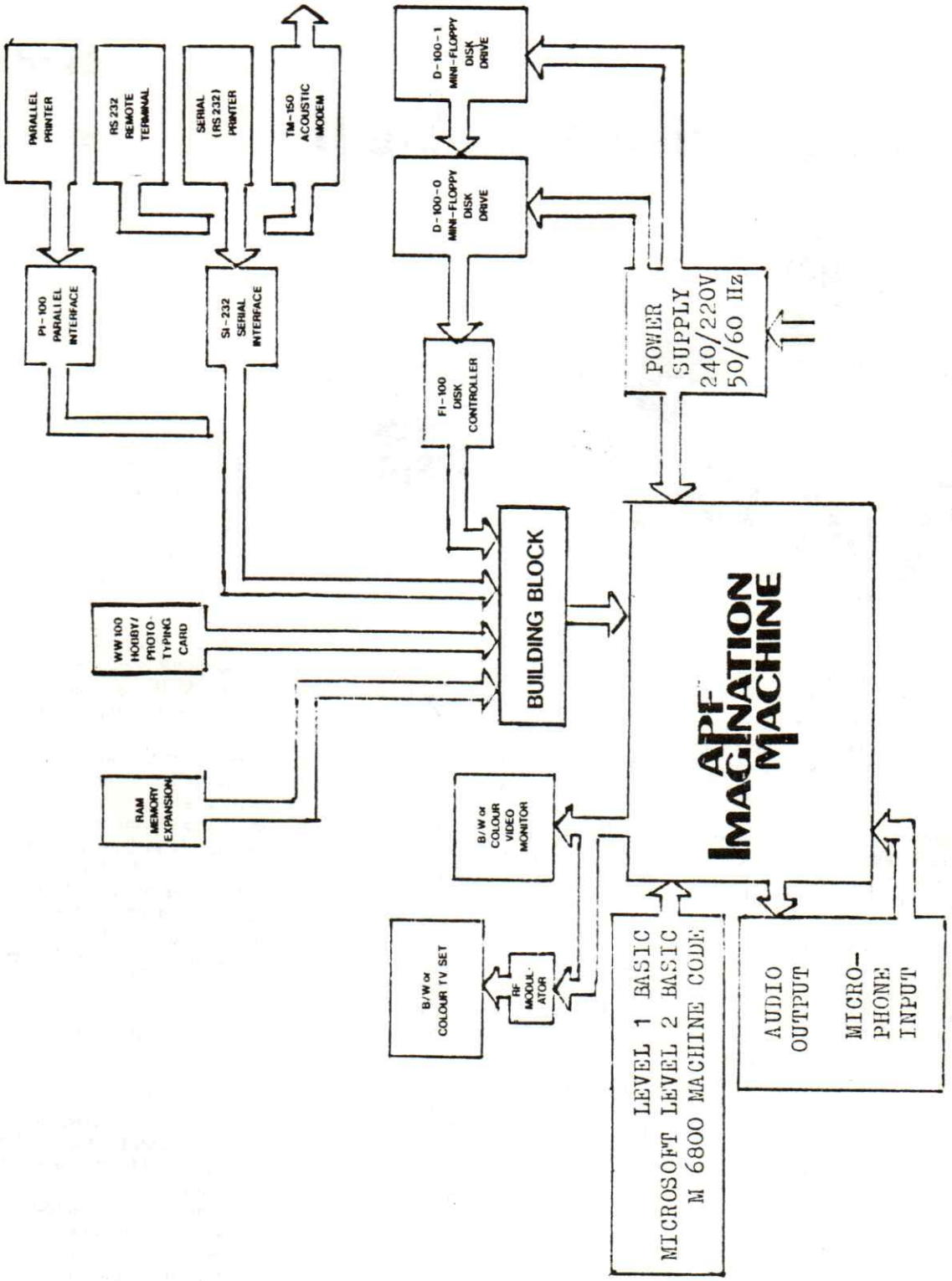
Another major screen and print feature is the PRINT USING command which allows the user to print numbers or strings in a specified format (leading dollar sign, two decimal places, etc.).

Another handy print feature is the ability to specify the line printer with the statement PRINT=1. To print on the screen, PRINT=0. This seems more flexible than the LPRINT command in other Basics. To list a program on the line printer one need only type in immediate mode: PRINT=1 and LIST.

Other features of APF Basic include an EDIT statement which allows the user to change a portion of a line in a Basic program. PEEK and POKE allow examination and alteration of values in memory. Several built-in machine language routines may be accessed by means of CALL statements, for example, CALL 17046 clears the screen to black while CALL 17026 creates a combination whistle/beep.

Basic has a range of ±999,999,999.9999. Plenty of accuracy for most small business.

EXPANSION SYSTEM ROLA-APF




**THE IMAGINATION MACHINE (IM-II)**

Exciting, personal, home computer for home management, entertainment and education. User-programmable in BASIC or uses APF software. Features 11K RAM, 14K ROM. 53-key keyboard, 32 characters x 16 line screen format, alpha numerics in up to 8 colors. Built-in sound synthesizer, built-in dual track cassette tape deck, expandable up to 75K RAM. Uses existing APF plug in peripherals. Single stroke entry of basic commands.

**TVM-10 VIDEO MONITOR**

- 10MHz band width
- 9" black and white video monitor
- Input impedance 75 ohms
- 9" L x 9" H x 9½" W
- Designed for monitoring computers, closed circuit TV and video tape recorders

**SI-232 SERIAL INTERFACE CARTRIDGE**

The SI-232 serial interface cartridge is used to connect most RS-232 serial devices to the Imagination Machine thru the Building Block. Eight selectable baud rates, 110 to 9600.

**PI-100 PARALLEL INTERFACE CARTRIDGE**

The PI-100 interface cartridge is used to connect most parallel devices to the Imagination Machine thru the Building Block.

**FI-100 MINI FLOPPY DISK INTERFACE**

**MINI-FLOPPY DISK D-100-0**

- 72,000 bytes per singlesided diskette.
- Standard SA400 BUS compatible
- Built-in power supply
- Standard IBM formatting
- Disk commands and functions built into basic interpretor cartridge

**MINI-FLOPPY DISK D-100-1**

- 72,000 bytes per singlesided diskette.
- Standard SA400 BUS compatible
- Built-in power supply
- Standard IBM formatting
- Disk commands and functions built into basic interpretor cartridge